REMARKS

Claims 1-18 are pending. Claims 1-18 are rejected by this Office Action.

The Applicants acknowledge the withdrawal of the objection to claim 1) and the corrections in reconsidering the previously filed IDSs.

Newly Cited Prior Art

The Office Action has cited the following additional prior art in rejecting some of the claims: "A Client-Side Stub Interpreter," August 1994 (Kessler) and "Industrial Applications of Distributed AI," November 1995 (Chaib-draa).

Kessler

Kessler discloses a client-side stub interpreter for supporting an operating system (Spring). Spring is a research prototype operating system that is designed to support distributed object-oriented programming. Applications that support services are made available through IDL interfaces. Client-side stubs support methods that a client program can call to invoke operations on objects defined by the IDL interfaces. A client-side stub method marshals any arguments for a corresponding operation and unmarshals any results. (According to the definition obtained from www.webopedia.com, data marsalling is the process of gathering data and transforming it into a standard format before it is transmitted over a network so that the data can transcend network boundaries. When the data is transmitted, the receiving computer converts the marshaled data back into an object.)

The disclosed client-side stub interpreter reduces the code size for a collection of methods by sharing common parts of the code and interprets a description of the variable parts. A typical kind of variability among client-side stub methods corresponds to numbers, nodes, and types of the parameters to the methods. Basic types can be marshaled and unmarshaled by the client-side stub directly. Kessler discloses a table of pointers to the marshal or unmarshal methods needed by a collection of IDL interfaces. Indices are stored into the table in the parameter descriptions.

Chaib-draa

Chaib-draa provides a survey of industrial applications of distributed artificial intelligence (AI) and provides several examples. The associated applications use distributed interpretation and distributed planning with intelligent sensors. One example relates to a Pilot's

Associate program, which is implemented with a set of five individual expert systems, to help pilots of advanced fighter aircraft.

Claim Rejections – 35 U.S.C. § 103

Claims 1, 4-6, 8-10, 13-15, and 17-18 are rejected under 35 U.S.C. 103(a) as being obvious over International Patent WO 97/44766 (Cook) in view of publication "A Client-Side Stub Interpreter" (Kessler).

The combination of Cook and Kessler fails to even suggest the feature of "managing information flow utilizing a **table of components**, wherein each component encapsulates behavior and data necessary to support a related set of services." (Emphasis added.) The Office Action admits that (Pages 3-4):

> However, Cook et al doesn't explicitly teach (c) managing information flow utilizing a table of components, wherein each component encapsulates behavior and data necessary to support a related set of services

The Office Action further alleges (Page 4):

> Kessler teaches (c) managing information flow (page 66, section 5.2, paragraph 2, "The client-side stub ... the normal stubs"; page 100, left column, paragraph 1, "would have to ... the desired properties") utilizing a table a table (page 97, section 3.2, paragraph 1, "Basic types can ... next table slot") of components (page 94, section 1, paragraph 1, "Spring is a ... of other services"), wherein each component encapsulates behavior and data necessary to support a related set of services.

In the above allegation, a "service" of the Spring operating system corresponds to a "component." As disclosed by Kessler (Page 94, section 1, paragraph 1):

> The services of the Spring operating system itself are made available through IDL interfaces, presenting a consistent programming model.

Kessler further discloses (Page 97, section 3.2, paragraph 1. Emphasis added.):

> The encoding we chose is to build a table of pointers to the marshal or unmarshal methods needed by a **collection** of IDL interfaces, and store indices in that table in the parameter descriptions.

Kessler merely discloses a table of pointers that do <u>not</u> point to components. A pointer, as disclosed by Kessler, points to marshal or unmarshal methods that are utilized by a <u>collection</u> of IDL interfaces. Services (alleged as being components) are made available through IDL interfaces. Each pointer (index) corresponds to parameter characteristics (e.g., type and mode) that are associated with the collection of IDL interfaces. Multiple IDL interfaces (associated with

different services of the Spring operating system), for example, utilize common marshal or unmarshal methods with common parameter characteristics. Kessler fails to even suggest a table of components. Thus, the combination of Cook and Kessler does not suggest the feature of "managing information flow utilizing a table of components, wherein each component encapsulates behavior and data necessary to support a related set of services."

Claim 10 includes the similar feature of "logic that manages information flow utilizing a table of components, wherein each component encapsulates behavior and data necessary to support a related set of services" and is not suggested by the combination of Cook and Kessler for at least the above reasons. Moreover, claims 4-6, 8-9, 13-15, and 17-18 ultimately depend from claims 1 and 10. The Applicants request reconsideration of claims 1, 4-6, 8-10, 13-15, and 17-18.

Furthermore, regarding claims 4 and 13, the combination of Cook and Kessler does not even suggest the additional features of "instantiating a component from the table of components to analyze progress and determine appropriate feedback" (as claimed in claim 4) and "logic that instantiates a component from the table of components to analyze progress and determine appropriate feedback" because neither Cook nor Kessler teaches anything about a table of components. Similarly, the combination of Cook and Kessler does not even suggest the additional features of "instantiating a component from the table of components to evaluate options and present appropriate feedback to assist the student to achieve the goal" (as claimed in claim 5), "logic that instantiates a component from the table of components to evaluate options and present appropriate feedback to assist the student to achieve the goal" (as claimed in claim 14), "instantiating a component from the table of components to simulate a business application" (as claimed in claim 6), "logic that instantiates a component from the table of components to simulate a business application" (as claimed in claim 15), "instantiating a component from the table of components to interact with the student utilizing rule-based logic" (as claimed in claim 8), "logic that instantiates a component from the table of components to interact with the student utilizing rule-based logic" (as claimed in claim 17), "instantiating a component from the table of components to present a time based simulation" (as claimed in claim 9), " and "logic that instantiates a component from the table of components to present a time based simulation" (as claimed in claim 18).

Claims 2-3 and 11-12 are rejected under 35 U.S.C. 103(a) as being obvious over Cook in view of Kessler and in further view of publication "Industrial Applications of Distributed AI" (Chaib-draa).

Claims 2-3 and 11-12 ultimately depend from claims 1 and 10. Because Chiab-draa does not make for the deficiencies of Cook and Kessler, claims 2-3 and 11-12 are patentable for at least the above reasons.

Furthermore, regarding claims 2 and 11, the combination of Cook, Kessler and Chiab-draa does not teach or even suggest the additional features of "instantiating a component from the table of components to measure progress toward the goal" (as claimed in claim 2) and "logic that instantiates a component from the table of components to measure progress toward the goal" (as claimed in claim 11) because neither Cook nor Kessler nor Chiab-draa teaches anything about a table of components. Similarly, the combination of Cook, Kessler, and Chiab-draa does not even suggest the additional features of "instantiating a component from the table of components to interrupt and interview the student to obtain information to measure progress toward the goal and determine appropriate feedback" (as claimed in claim 3) and "logic that instantiates a component from the table of components to interrupt and interview the student to obtain information to measure progress toward the goal and determine appropriate feedback" (as claimed in claim 12).

Claims 7 and 16 are rejected under 35 U.S.C. 103(a) as being obvious over Cook et al in view of Kessler and further in view of USPN 5,727, 161 (Purcell).

Claims 7 and 16 ultimately depend from claims 1 and 10. Because Purcell does not make up for the deficiencies of Cook and Kessler, claims 7 and 16 are patentable for at least the above reasons.

Furthermore, regarding claims 7 and 16, the combination of Cook, Kessler and Purcell does not teach or even suggest the additional features of "instantiating a component from the table of components to interact with a quantitative analysis model to perform what-if-analysis" (as claimed in claim 7) and "logic that instantiates a component from the table of components to interact with a quantitative analysis model to perform what-if-analysis" (as claimed in claim 16) because neither Cook nor Kessler nor Purcell teaches anything about a table of components.

All objections and rejections have been addressed. Hence, it is respectfully submitted that the present application is in condition for allowance, and a notice to that effect is earnestly solicited.

Respectfully submitted,

Date: October 7, 2005

Kenneth F. Smolik
Registration No. 44,344
BANNER & WITCOFF, LTD.
10 S. Wacker Drive, Suite 3000
Chicago, IL 60606-7407
Telephone: 312-463-5000
Facsimile: 312-463-5001